The Digital Signal Processing Applications Newsletter

DSPatch

Number 35 Spring 1996

New DSP Generation Optimized For Concurrent Processing

In this issue

ANALOG DEVICES

Anticipating a new generation of concurrent architecture digital signal processing applications, Analog Devices has introduced the ADSP-21csp01—first of the company's products to incorporate a new 16-bit fixed-point core. Its design specifically targets computer and telecommunications applications that must handle several signals simultaneously. In voice-over-data modems, cellular-phone basestations, and computer telephony systems, the ADSP-21csp01's innovative architecture will improve DSP throughput while reducing end-product chip count and time to market.

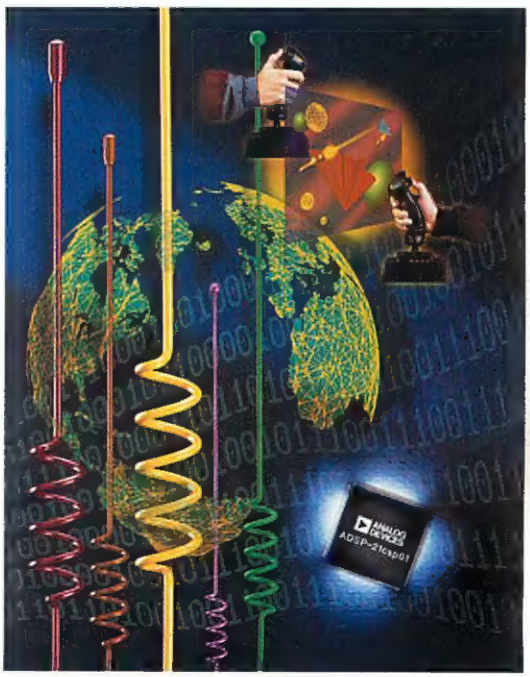
Meeting Designers' Demands

System designers have long sought to take advantage of ever-faster DSPs by processing multiple tasks or multiple signals on one device. Unfortunately, most DSPs lack the necessary data-throughput and I/O capabilities. The ADSP-21csp01 design addresses this precise requirement by combining a highly-parallel, 50 MIPS DSP core that can execute 550 million operations per second (MOPS) with a parallel DMA port, a parallel memory port, two multichannel serial ports, and low-latency interrupt servicing featuring fast single-cycle (20ns) task switching.

The device core contains 96 on-chip registers—64 addressing registers and 32 arithmetic registers, including two sets of multiply registers. The large number of registers allows the device to hold multiple data sets, so that data can switch easily in and out of the computational pipeline with a minimum of software overhead. For example, a compiler need not generate extra instructions to save register values to external memory and restore them later. This architecture improves computational performance over conventional DSPs and assures optimum data flow.

On-Chip Peripherals Increase Throughput

To minimize interruption of core processing functions and maximize data flow in and out, the ADSP-21csp01 contains several functions



New Look

Welcome to the first issue of the newly designed DSPatch. Our introductory issue hit the stands back in 1986. As ten years have passed, so much has changed in the world of technology and digital signal processing that we decided to change as well—just our look, not our commitment to publishing a digital signal processing applications newsletter that is technically informative.

The newsletter still contains the same sections, only now they are more recognizable as separate entities:

DSP In Use—These articles describe end products designed around Analog Devices DSPs and illustrate the benefits of Analog Devices architecture.

Q&A—The question and answer section documents solutions to assorted design problems that have been encountered by some of our customers.

Up To Date—This section contains various news articles concerning recent developments, new releases, and product updates.

Third Parties—This area highlights third party suppliers who can help you get through your design and debug cycle quickly.

C Tips—This section is an ongoing look at programming with C. Each issue we focus on one particular area of programming a DSP in this high-level language.

If you have any ideas about the newsletter or ways we can serve you better, let us know. Contact us by fax at (617) 461-3010 or by email at cpd_applications@analog.com. ❖

Come Visit Us at DSP[X] '96!

Do you find it increasingly difficult to keep abreast of the latest developments in signal processing tools and technology? Come to DSP[X] for an unparalleled opportunity to see what others are thinking, designing, and implementing in all areas of signal processing. The conference will be held on March 11-14 at the San Jose Convention Center in San Jose, California.

Analog Devices, with over thirty years in signal processing, is a technology leader. Visit us at DSP[X] and see why Analog Devices' DSPs are becoming the processors of choice for more and more system designers. Analog Devices' Exhibit Booth (# 1202) provides a great opportunity for you to get an in-depth look at our fixed-point and floating-point DSP families. You can also explore the capabilities of the various software and hardware development tools and talk face-to-face with DSP application engineers. See demonstrations of the newest DSP processors and get the most up-to-date information of the first member of the newest fixed-point DSP family, the ADSP-21csp01. Furthermore, you can find out what's new in the area of third party providers that can help you through the design and debug stages quicker.

DSP[X] includes:

- New, expanded Signal Processing Applications Conference
- New Executive Management Forum
- New Product Presentation Forum
- Attendee Solutions Center
- "Hands on" product demo area
- Keynotes by industry visionaries

Check out the DSP[X] worldwide web site (http://www.dspx.com:2000) for more information. You can also call (203) 840-5652 or email at dspx@reedexpo.com. ❖

Using DSP

Mercury Computer's RACE Series

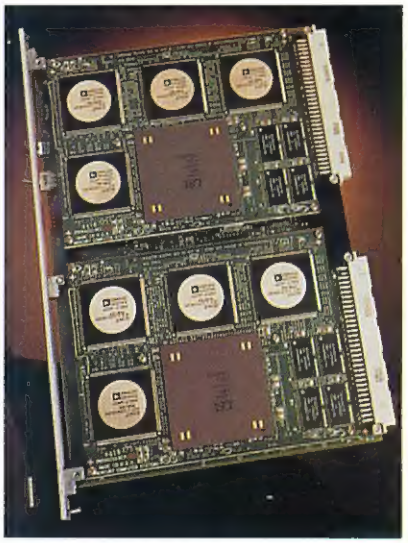
Mercury Computer Systems offers products based on both the Analog Device ADSP-21060 and ADSP-21062 SHARC DSP processors. The ADSP-21060/62 is among the higher performing programmable floating-point processors in the industry. By integrating the SHARC into Mercury's RACE® multicomputing environment, Mercury offers developers the capability of deploying hundreds of SHARCs and RISC processors in a unified system.

Hardware scalability is complemented by Mercury's Multicomputer Software Backplane™ system software. Every processor in a RACE® system runs (at least) a subset real-time POSIX kernel upon which layered multicomputer software tools help the developer solve multiprocessing applications. For example, all nodes communicate with the Interprocess Communication System and are tuned with visual gdb and SuperVision debuggers. Also multiprocessing algorithms are developed with the standard Parallel Application System (PAS). Proven with routine installations of over 190 processors, software from Mercury and its RACEware™ partners provide the serious system developer with the leading environment for real-time, scalable multicomputing.

Mercury integrates the SHARC DSP processor into a Compute Node (CN) which consists of one or more processors, DRAM memory, and a Mercury designed ASIC. The CN ASIC provides high-performance functions needed for real-time multicomputer DSP. The CNs are designed to fit as daughtercards on either Mercury's RACE-MCH6' (VME64 6U), or RACE-MCH9' (VME64 9U) motherboard. The CNs are connected to each other with the high-performance, industry standard RACEway Interlink switching fabric (ANSI/VITA 5-1994).

Mercury currently offers two SHARC CN configurations targeted for different applications—S2T16B and the S1D64B. The S2T16B/8B is designed for maximum processor density. It implements six SHARC processors, arranged in two independent CNs of three processors each. Each CN on the daughtercard has either 8 or 16 Mbytes of DRAM that is shared by the CN's three SHARC processors. The S1D64B is designed to maximize the amount of memory per SHARC processor. It consists of a single CN containing two SHARC processors and 64 Mbytes of DRAM. This card is an ideal fit for memory-intensive applications.

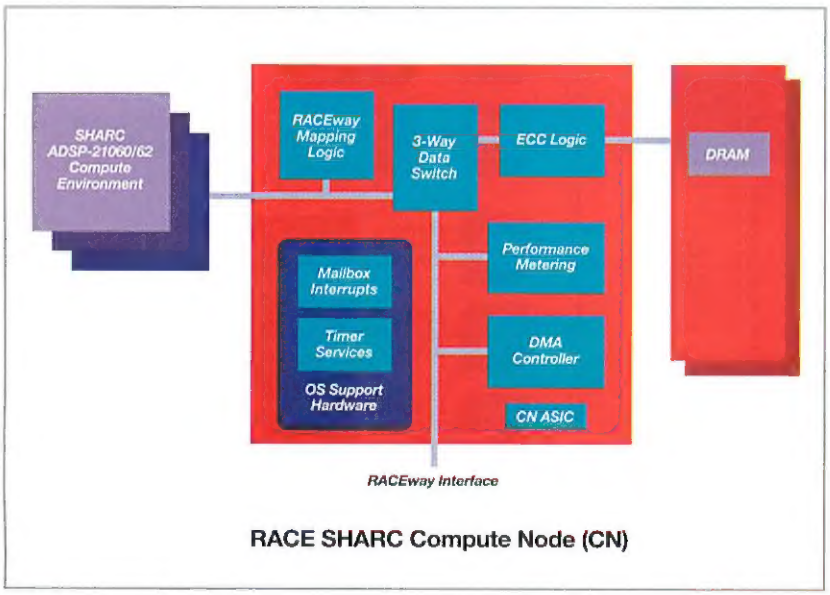
With these daughtercards, Mercury can provide scalable configurations going from two up to 12 SHARC DSP processors on a single VME64-6U board and from two up to 42 (possibly up to 48 depending on final production power testing) on a single VME64-9U board. Mercury provides systems which can scale to several hundred processors with a very small physical and power footprint.



The RACE® Series with the SHARC is ideally suited for embedded DSP applications—such as airborne reconnaissance, ship borne sonar acoustic processing, and medial image reconstruction, where cost, space, and power are at a premium.

You can read more about these and other Mercury products on our Web page at http://www.mc.com.

Mercury Computer Systems, Inc.
199 Riverneck Road
Chelmsford, MA 01824
Tel: (508) 256-1300
Fax: (508) 256-3599 ❖



RACE SHARC Compute Node (CN)

Pinnacle Micro Develops 4.6 Gigabyte Optical Disk Drive

by Keith Holstine, PhD.
Pinnacle Micro R&D Center

The quest for higher density storage devices continues as applications get larger and data files for some of these applications, such as multimedia, get immense. In response to this need, Pinnacle Micro has developed the Apex Optical Hard Drive. This drive is a high performance Magneto-Optical (MO) disk drive system which employs Analog Devices' ADSP-2171 as the core for all servo system functions. The drive conforms to the 5.25" half-height form factor and weighs about three pounds. The media is removable and can store up to 4.6 Gbytes of user data per cartridge.

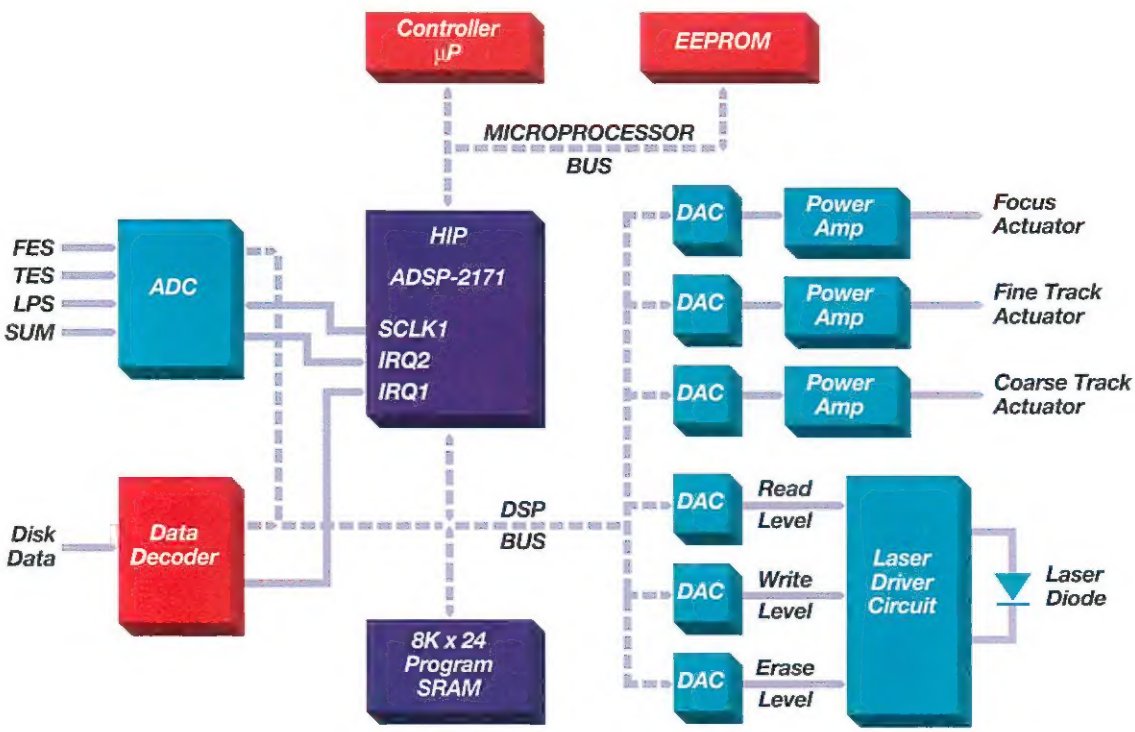
The system is designed to compete with magnetic hard drives for capacity, performance, and cost. The 4.6 GB user data capacity, maximum data transfer of 6 MB/sec, and 19 ms average seek time make the Apex a very competitive product. The cost per megabyte is only 17 to 37 cents, depending on quantities, for the Apex drive and first disk cartridge. The cost per megabyte is about 4 cents for each additional disk cartridge.

ADSP-2171 Servo Control

The ADSP-2171 is used to focus the laser, perform seek operations to track and sector locations, and control laser power levels. For focus and tracking, three servo actuators are controlled simultaneously by the ADSP-2171. The positioning error allowed for both focus and tracking is less than 1 micron.

A single EEPROM connected to the microprocessor bus contains both the controller microprocessor and DSP servo system firmware for the Apex optical drive. The firmware for the servo system is downloaded into the DSP by the controller microprocessor at powerup using the ADSP-2171 Host Interface Port (HIP). The HIP download method allows for easy field upgrades of DSP servo system firmware, directly through a SCSI download program. The HIP is also used to efficiently pass commands and parameters from the controller microprocessor to the DSP, and return data and status from the DSP to the microprocessor.

Four servo signals are required to control the focus and tracking actuators. The Focus Error Signal (FES), Tracking Error Signal (TES), and a reflected light intensity signal (SUM) are obtained from laser light reflected from the surface of the disk. In



ELECTRICAL SYSTEM BLOCK DIAGRAM

addition, a Lens Position Signal (LPS) is generated that measures the position of the lens with respect to the center of its motion in the tracking direction.

FES is used to control the focus actuator and keep the laser spot focused on the reflective surface of the disk. TES is used to control the fine tracking actuator and keep the laser spot following the track on the disk. The coarse tracking actuator is controlled by LPS. The coarse tracking actuator follows the lens, and keeps the lens positioned near the center.

The DSP is also directly interfaced to the data decoder IC. The data decoder provides track and sector information read from the disk to the DSP. The data decoder generates an interrupt each time a new track and sector value is read. This information is used to perform seeks to any desired location on the disk to access user data.

Firmware Description

The DSP servo system firmware is written to perform as an interrupt-driven state machine. The ADSP-2171 is particularly well suited for this type of firmware structure, due to its short interrupt latency (~ 3 instruction cycles). The ADSP-2171 also provides for automatic nesting of interrupts. This feature is used in the Apex DSP firmware.

After the DSP firmware has been boot loaded using the HIP, the ADSP-2171 is initialized and the code in both the internal and external program memory is then verified using a checksum. The internal data memory is also tested. Then a system diagnostic is performed that tests all other servo hardware. After all testing is complete, interrupts are enabled.

Each time the ADC generates a conversion complete interrupt (IRQ2), the DSP reads and saves the sample value. The sample type (FES, TES, LPS, or SUM) is determined by reading the state of the two ADC multiplexer control lines. The sample value is immediately used for control before returning from the interrupt service routine. For example, as soon as a FES sample is received, the focus error data is processed using Infinite Impulse Response (IIR) filters. The IIR filters are implemented using difference equations to provide the lag, lead, and notch filtering required to compensate the focus servo loop. The output value calculated using the firmware filtering is written to the focus DAC, and then the code returns from the interrupt. The control for the fine and coarse tracking servo loops is accomplished in the same way as the focus servo, using individual interrupt service routines for each loop. Each of the servo interrupt service routines must be kept short enough so that it is completed before the next servo sample is taken by the ADC.

"The ADSP-2171 is particularly well suited for this type of firmware structure, due to its short interrupt latency structure."

The ADSP-2171 has a secondary register set for quick context changing during interrupts. The servo firmware uses the secondary registers while servicing IRQ2, since these routines are the most register intensive, and also most frequently executed.

The DSP firmware also responds to interrupts from the data decoder (IRQ1). Each time this interrupt is generated, the DSP reads the current track and sector number from the data decoder, and saves the location information. This information is used to seek to a specific location, and also to "hold" on a given track number when the drive is idle.

The laser power for reading data is usually set to a constant level by the DSP firmware. On the other hand, the laser power levels for writing and erasing data are changed as a function of the location of the spot on the disk. Higher power levels are needed to write and erase data near the outer diameter of the disk, than at the inner diameter. This is primarily due to the fact that the disk has a higher linear velocity near the outer diameter.

Whenever the DSP is not servicing either servo control interrupts (IRQ2) or data decoder interrupts (IRQ1), the firmware continuously polls the HIP interface to see if a new command has been received from the controller microprocessor. The HIP provides hardware register status bits that allow the DSP to determine if any of the HIP data registers have been written (or read) by the microprocessor.

The DSP firmware uses one of the flag output pins of the ADSP-2171 to indicate that a command is being processed. In addition, another one of the flag output pins is used to indicate attention conditions. An attention condition is a warning to the controller microprocessor of any abnormal DSP condition or problem. The specific problem status is read by the microprocessor using the HIP. This type of attention flagging allows the DSP to have true bidirectional communication with the controller microprocessor.

The ADSP-2171 has many additional features that make it the perfect choice for DSP-based servo system development. The ADSP-2171 has proven to be a very capable DSP product for the high performance servo requirements of the Apex Optical Hard Drive.

For more information about Apex contact:

Pinnacle Micro
19 Technology
Irvine, CA 92718
Tel: 1-(800) 553-7070 ❖

Anatomy of an ADSP-2100 Family C Program

This C tip explains in detail a basic ADSP-2100 Family C program. This C routine provides a simple shell for use with the ADDS-21XX-EZ-LITE. While the EZ-KIT Lite comes with software, it does not include the C compiler. So this C tip assumes that you have the complete version of the Analog Devices software (ADDS-21XX-SW-PC). The current version of the software is release 5.1.

Our main C program is CTIP35.C. As you can see from the listing, individual lines have been numbered. The lines are explained in the corresponding sections below. CTIP35.C, when run on the EZ-KIT Lite, takes the input from the AD1847 stereo codec called LEFT_IN and RIGHT_IN and loops by writing to LEFT_OUT and RIGHT_OUT. You can add on to this simple talk-through program by modifying the input values before writing to the outputs.

The file CTIP35.ZIP contains all the system files needed to test this application on your own EZ-KIT Lite system. CTIP35.ZIP is available on the Analog Devices BBS or FTP. Files included in the ZIP file are:

CTIP35.C
SIGNAL.H,
2181_HDR.DSP
TALK_47.DSP
BUILD.BAT
2181.ACH

The file TALK_47.DSP is an assembly routine that is called by CTIP35.C and initializes the interface between the ADSP-2181 and the AD1847. The file SIGNAL.H is a header file that contains macros for handling interrupts. 2181_HDR.DSP contains a modified runtime header for the ADSP-2181. BUILD.BAT is a batch file that assembles 2181HDR.DSP and then invokes the C compiler and all its switches. And finally 2181.ACH is an architecture file that can be used with the EZ-KIT Lite.

For more information on C coding with the Analog Devices' DSPs, refer to the ADSP-2100 Family

```
CTIP35.C
  /* CTIP 35 */
  /* this program can be used as a shell for C programs running on the
     EZ-KIT Lite board or as an basic example of C code */
3 #include <signal.h>
2 #define  DM_Wait_Reg     *(int *)     0x3ffe
5 extern init_1847();
3 void new_sample();
4 asm(".var/dm/ram/circ rx_buf_[3];            /* Status + L data  + R data */\n"\
4     "\t.var/dm/ram/circ tx_buf_[3];          /* Cmd + L data + R data */\n"\
4     "\t.init tx_buf_: 0xc002, 0x0000, 0x0000; /* Intially set MCE */\n"\
4     "\t.global rx_buf_, tx_buf_;");
1 volatile int left_in, right_in;
1 volatile int left_out, right_out;
  int stat_flag;
  void main()
  {
5       init_1847();                  /* initialize 1847 interface */
4      asm("set fl1;")                /* set LED on EZ-KIT Lite */
2      DM_Wait_Reg=0x0fff;
3      interrupt(SIGSPORT0RECV, new_sample);
6      while(1){
6      asm("idle;");                  /* wait for an interrupt */
6      right_out=right_in;            /* loop back for talk-thru */
6      left_out=left_in;
        }
  }
3 void new_sample()
  {
4      asm("ena sec_reg;");
4      asm("ax0 = dm(rx_buf_+1);");   /* receive sample */
4      asm("ax1 = dm(rx_buf_+2);");
4      asm("dm(left_in_) = ax0;");    /* write to memory */
4      asm("dm(right_in_) = ax1;");
4      asm("ax0 = dm(left_out_);");   /* read output sample */
4      asm("ax1 = dm(right_out_);");
4      asm("dm(tx_buf_+1) = ax0;");   /* transmit to codec */
4      asm("dm(tx_buf_+2) = ax1;");
4      asm("dis sec_reg;");
  }
```

C Tools Manual, the ADDS-21XX-SW-PC release note, or previous C Programming With DSP articles.

1) Extensions to the ANSI C Compiler

The Analog Devices GNU-based C compiler conforms to the ANSI C standard. However, there are additions or extensions to the ANSI C language that are specific to the ADSP processors. The instruction:

```
volatile int left_in, right_in;
```

is an example of an extension to the `int` data type. Using volatile will force the DSP to place the variables LEFT_IN and RIGHT_IN into memory locations—as opposed to manipulating the data in registers only. Another extension to ANSI C is the `pm` data type. Using the data type:

```
int pm coeff[10];
```

will create a buffer in the DSP's program memory for storing coefficients. Without the pm extension all data will be placed in data memory.

2) Accessing Memory-Mapped Registers in C

The DSP reserves a section of internal memory for configuration registers. Configuring the serial ports, internal timer, IDMA, BDMA, memory wait states, etc. is accomplished by writing to these memory-mapped registers. To accomplish this in C, use a `#define` command to create a label or pointer for the memory location. In our CTIP35.C program we're initializing the wait states for the IO memory space using:

```
#define DM_Wait_Reg *(int *) 0x3ffe.
```

Then in the code, you can write to that memory location using:

```
DM_Wait_Reg=0x0fff;
```

3) Handling Interrupts

The DSP is an interrupt-driven processor. In order to program interrupts in C, it would be helpful if you understood what interrupts are available on your DSP of choice. The ADSP-2181 for example has interrupts for serial ports, external signals, timer, powerdown, etc. You can use the CTIP35.C as a shell. The next section describes how we've configured the interrupts and how to configure interrupts for your own system.

The first step to handling interrupts is to include the header file SIGNAL.H in your program using the line:

```
#include <signal.h>
```

SIGNAL.H contains macros for each interrupt of each ADSP-2100 family processor.

Note: SIGNAL.H is included with your software tools. However, even if you have the latest software, you may not have the latest version of the file. It is available for downloading from our BBS or FTP site or is included with the files if you download CTIP35.ZIP.

For CTIP35.C, we need to set up the interrupt for Serial Port 0 Receive. The macro has the following syntax:

```
interrupt(SIGSPORT0RECV, new_sample);
```

This line identifies the function `new_sample` as the interrupt handler for serial port 0 receive interrupts.

4) Embedding Assembly Instructions in C Code

Admittedly, as soon as you add in-line assembly code you begin to sacrifice some of your C code's portability. However, there are some instructions that you can only perform in assembly. Creating a buffer can be done just as easily in C as in assembly. However, only using the assembly directive:

```
asm(".var/dm/ram/circ rx_buf_[3];");
```

you can create a circular buffer. The IDLE instruction and system register accesses such as the IMASK also need to be done with assembly code. The general format for embedding an assembler instruction is:

```
asm("assembly code;");
```

To include several assembly lines of code, use the format:

```
asm("first instruction;" \
"second instruction;");
```

The backslash (\) has to be the last character on the line (including comments). You can also use \n and \t for creating line returns and tabs in you listing file.

5) Calling DSP Programs/Routines

Another way to add assembly code to your C routine is to place the DSP code in a C callable function. The program TALK_47.DSP contains code to initialize the AD1847 codec. TALK_47.DSP contains the routine `init_1847()` which is called from C. An underscore must be added to labels and variables that are initialized in C and accessed in assembly. The DSP label in TALK_47.DSP is `init_1847_`.

6) Handling Real-Time Environments

Most DSP programs are based on external signals. This program waits until a Serial Port Receive interrupt occurs and then performs the loopback. The WHILE instruction is used to create an infinite loop. The IDLE assembly instruction makes the processor wait for an interrupt. The catch is your code has to complete before the next interrupt would occur. For example, if you receive data from the AD1847 every 16 kHz you'd have 1/16 kHz or 62.5 microseconds between interrupts. Using a 33 MHz ADSP-2181, this equates to 2062.5 DSP cycles.

BUILD.BAT

```
7 asm21 2181_hdr -c -2181
7 g21 -a 2181 -o ctip35 -I.
   -mreserved=i2,i3 -runhdr
   2181_hdr.obj ctip35.c talk_47.dsp
   -g -save-temps - mlistm
```

7) Compiling Your Program

Before using the g21 command to compile the CTIP35.C program, you need to assemble your new ADSP-2181 runtime header. Use the -c switch to make the 2181_HDR.OBJ file case sensitive. The switches that are only needed for debug are the -g, -save-temps, and -mlistm.

After using the batch file to compile your C code, you can download the executable CTIP35.EXE to the EZ-KIT Lite using the EZ-KIT Lite monitor program that runs under windows.❖

Q Using Data Tables in ADSP-2181 Boot ROM

I would like to take advantage of the BDMA feature of the ADSP-2181. Specifically, I would like to have data tables residing in the EPROM that I can dynamically load into the internal memory of the ADSP-2181 during program execution. I know how to create a PROM file for the code and data for my program using the development tools but I am not sure how to get a data table into the PROM. Can you help me?

The BMDA Feature of the ADSP-2181 Allows Easy Access to Data Tables

A *The ADSP-2100 Family Development Tools allow you to create a program which includes code and data. The Prom Splitter will format the executable to create a file that can be downloaded to a PROM programmer. This PROM Splitter also appends some boot loader code to your program. The PROM Splitter assumes that all your code and data is to be loaded into the internal memory of the ADSP-2181 at power-up boot. If your data table is to be loaded during boot, there is nothing special that you have to do.*

If you would like to include a data table, such as a lookup table, in the EPROM that can be transferred into the ADSP-2181 during runtime, you must manually program this into the EPROM. Since most PROM programmers allow you to perform multiple programming passes, you can first program the EPROM with your executable. The next step involves loading your data file into the PROM programmer and specifying a destination location of the EPROM. There are a total of 4 Mbytes of usable space specified with 22 address bits. You must make sure that the data table is placed in a blank area of the EPROM and does not overlap with the executable. You must also make note of this EPROM address since it will be needed when you program the BDMA transfer.

When it is time to transfer the data table from the EPROM into the internal memory of the ADSP-2181, you must initiate a BDMA transfer. The process is as follows:

1) *Write the 14 least significant external EPROM address bits into theBEAD register, DM(0x2FE2).*
2) *Write the 8 most significant external EPROM address bits into the BMPAGE field of the BDMA control register, DM(0x3FE3).*
3) *Set the BTYPE, BDIR, and BCR fields of the BDMA control register to the appropriate values for your application. For example, if you are loading the data table into the ADSP-2181's data memory and you want the ADSP-2181 to continue executing instructions while the data table is loaded in the background, set BTYPE to '01'; set BDIR to '0'; and set BCR to '0'.*
4) *Write the internal ADSP-2181 destination address into the BIAD register, DM(0x3FE1).*
5) *Write the word count into the BDMA word count register, DM(0x3FE4).*
6) *The BDMA transfer begins when the word count is written.*
7) *A BDMA interrupt will occur when the BDMA transfer is complete.* ❖

Q ADSP-2181 Flag Pins

I am having difficulty getting the flag pins to respond to my program. How do I write the programmable flag data? Is the PFDATA register the same as the "PF" pins on the schematic?

A *The ADSP-2181 has eight general purpose programmable input/output flag pins. These do correspond to the PF0–PF7 pins shown on the schematic of the device. There are two memory-mapped registers in the ADSP-2181 that are used to program the flag (PF) pins—the PFTYPE register DM(0x3FE6) and the PFDATA register DM(0x3FE5). The least significant seven bits of the PFTYPE register must first be set to specify the direction of the flag as input or output. The least significant seven bits of the PFDATA register correspond to the state of the flag pins. The example below configures PF0 as an output flag and sets the flag pin high.*

Programmable Flags Can Be Used for Input or Output

```
AX0 = 1;
DM(0x3FE6) = AX0; /* Configure PF0 as an output */
DM(0x3FE5) = AX0; /* Set PF0 to 1 */
```

Keep in mind that the upper bits of the PFTYPE register are used for other functions. You should refer to page 14 of the ADSP-2181 data sheet.

You can also set these registers by using indirect addressing. Here is an example of how to do this.

```
I0 = 0x3FE6;        /* Load address of PFTYPE into I0  */
M0 = -1;
L0 = 0;
AX0 = 1;
DM(I0, M0) = AX0;   /* Configure PF0 as an output */
DM(I0, M0) = AX0;   /* Set PF0 to 1 */
```

To set, reset, or toggle flag values you can use the bit manipulation instructions of the ADSP-2181. These are detailed on pages 15 -29 of the ADSP-2100 Family User's Manual. The following example uses the TGLBIT function. ❖

```
AX0 = 1;
DM(0x3FE6) = AX0;       /* Configure PF0 as an output */
DM(0x3FE5) = AX0;       /* Set PF0 to 1 */
AR = TGLBIT 0 of AX0; /* Toggle Bit 0 */
DM(0x3FE5) = AR;        /* Set PF0 to toggled value */
```

Analog Devices on the World Wide Web

You can now visit Analog Devices on-line via the World Wide Web at www.analog.com. Our web site helps us get timely and accurate technical information to you faster than ever. Complete data sheets for all announced products can be downloaded directly to your PC or printer. In fact, our site was recently chosen by EE Times readers as one of the best engineering sites on the Web. Interactively access information 24 hours a day and browse product literature, device specifications, technical support information and much more.

The site features capabilities like a powerful search engine to locate your topic of interest and a cross-reference guide to locate a specific part number. A convenient site map is provided to help you navigate through our Web site.

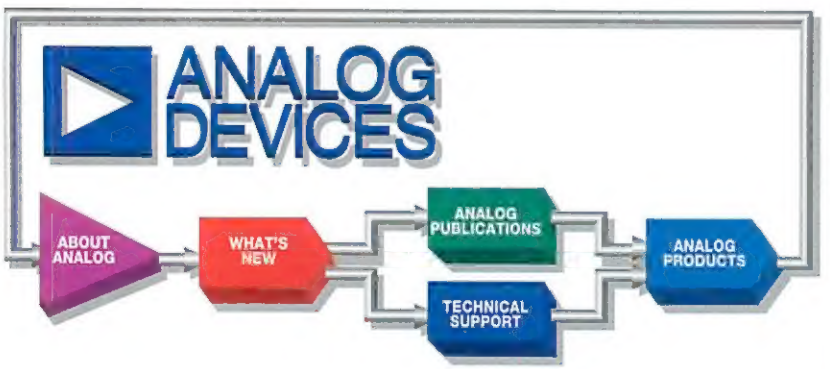
Four primary sections simplify browsing:

About Analog

Learn about Analog Devices by accessing the Corporate Profile and Earnings Reports. You can also review job opportunities within Analog Devices.

What's New

Under this topic you can find the latest additions to the Web site. Check here for the latest news, press releases, updated trade show schedules, and other happenings.



Analog Products

Check out our Signal Chain section. Using the signal chains you can find a collection of typical applications, with block diagrams and suggested parts that are well suited to that system. Use our Selection Tree section to find products you are looking for. These trees diagram the key parameters and choices; using them allows you to find parts within major product types.

Publications

Find the latest issues of product press releases, whitepapers, product documentation (data sheets, manuals), newsletters such as DSPatch, brochures and seminar materials. ❖

European Technical DSP Support Center Established

Getting the applications support and product information you need to design with Analog Devices' DSPs just got easier for our growing number of European customers. The European DSP Customer Support Group located in Munich, Germany has been established to provide technical support to European DSP customers. The group can be accessed during usual business hours, 9:00 A.M. to 5:00 P.M. MET (Middle European Time).

Multi-lingual engineers provide applications support to English-, German-, French-, and Spanish-speaking customers. The group plans to add Italian-speaking engineers as

well. Timely, accurate information is assured through a direct connection to the factory using Lotus Notes. The Lotus Notes system routes action items to the appropriate product, design, or application engineering experts within Analog Devices, solving problems quickly and efficiently.

What Services Are Available?

The European DSP Customer Support Group offers a fax hotline and an email address for all customers. The group has a goal to respond within 48 hours to all inquiries. A European Bulletin Board System (BBS) is also available. You can download the latest software updates, information, and examples from the BBS.

Additional Capabilities

The European DSP Support Group also coordinates DSP training and workshops for all of Europe and maintains the DSP training lab in Munich. Hardware and software development tools from Analog Devices and many third parties are running in the lab, enabling better simulation of problems and faster identification of solutions.

What Next?

Within the next month, the European DSP Customer Support Group hopes to add support for sales and marketing inquiries, such as new product information and monthly products status updates.

Access ADI - Europe

Please don't hesitate to use the new European support services and your feedback is welcome and greatly appreciated.

DSP Applications
Analog Devices GMBH
Edelsbergstr. 8-10
80686 Munich
Germany
Fax: ++49-89-57005-200
Email: dsp.europe@analog.com
BBS: ++43-1-8887656 ❖

ADSP-21csp01 Designers' Demands

(continued from page 1)

that normally reside on peripheral chips. Two bidirectional serial ports, each supporting 2 to 32 TDMA channels and Direct Memory Access (DMA), can send and receive data at 25 Mb/s. The resulting four paths provide a total transfer capability of 100 Mb/s. A 16-bit internal DMA (IDMA) parallel port connects this DSP to other processors and to system buses. A DMA controller supports five channels between the serial ports or between the IDMA port and device memory. The serial ports contain their own FIFO buffers, which allow converters and other devices to move data into and out of DSP memory, interrupting program flow only when the transfer is complete.

ADSP-21csp01 Optimized For High-Level Language Software Development

Analog Devices has tuned the ADSP-21csp01's architecture to maximize efficiency during execution C-based software algorithms. Recent architectural advances, for example, have improved stackhandling, memory allocation, data flow, and computation activities, the areas that most affect C programs for DSP.

DSPs generally incorporate a Harvard-type architecture, fetching a data word from one memory space and another word, instruction, or coefficient from a separate memory space during each clock cycle. Although this arrangement supports two parallel I/O buses, it requires separate data spaces for each bus, restricting a C compiler's freedom to allocate memory. Therefore, to achieve maximum operational efficiency, C-language microprocessors rely on a Von Neumann design, which provides a single data space that the compiler divides up on-the-fly. This arrangement, however, limits data transfer to one I/O bus.

The ADSP-21csp01 core overcomes the Von Neumann limitation by defining a continuous memory space, but two I/O buses. Each bus still addresses a particular section of memory, but the compiler decides which section carries instructions and which contains data clock-cycle by clock-cycle, keeping careful track of how it apportions the space.

C programs typically consist of many functions and subroutines called by a main program. To manage the flow, the compiler sets up stack frames organized as linked lists. The ADSP-21csp01 device core's indirect data addressing allows it to retrieve values from the stack and send values to the stack without extraneous data-move instructions. The address generator can point to an array where some elements are also pointers, loading those pointers into address registers in a single cycle. Such arrays are common with subroutine call and return instructions, so this capability produces faster, more efficient software than a more conventional DSP architecture can provide.

The ADSP-21csp01 represents a new direction in digital signal processing. Its innovative core architecture will become an important ingredient in all of Analog Devices' future DSP products. As a result, system designers can finally achieve the exalted performance that concurrent processing applications demand. ❖

Using DSP

SHARCs Help SEGA Win the Game

Gentlemen, start your engines! You're on the grid at the Indianapolis 500, the green flag drops, and your foot hits the floor. You're strapped in for the ride of your life, racing to see who gets to the checkered flag first. You successfully maneuver around your opponents to reach the finish line. The crowd goes wild!

"Our new Model 2B-CRX has the capability of rendering 300,000 polygons per second, the highest performance computer graphics system in the industry."

Scenarios like this are everyday occurrences on SEGA Enterprises' new electronic arcade games. What's behind the clear, crisp picture are Analog Devices' SHARC chips. SEGA uses the ADSP-21062 digital signal processor, otherwise known as the SHARC, for processing the intensive mathematical calculations that give the games their distinctive features.

SHARC Improves SEGA Performance

According to Hiroshi Yagi, SEGA General Manager, AM Hardware, R&D Department, two SHARCs were used to replace SEGA's original system of five DSPs. "The SHARCs contributed to improving the overall performance of our new Model 2B-CRX, including cost, space and electrical performance. This model has the capability of rendering 300,000 polygons per second, the highest performance computer graphics system in the industry."

The many scenes, through which the game player drives, are broken up into a large number of very small polygons. As the player manipulates the steering wheel, gas pedal, break and gear shift, the system must provide real-time graphics processing response. The 40 MHz SHARC with 120 MFLOPS peak performance plays a key role in this graphics processing function. One of the two SHARCs is used as a geometry engine which manipulates polygons in real time to create the three-dimensional visual effects. A second SHARC is connected to a RISC processor as a floating-point accelerator. The host port of the ADSP-21062 simplifies the connection to the RISC processor.

In the INDY 500 game, an extremely authentic driving experience is created due to the high speed processing of the complex graphics. At the start of the game, various views are shown at different perspectives. A very realistic perception is created as though a camera is being aimed at the car from different angles. In reality, the views are created by a three-dimensional graphics simulation. During game play, you can select from several perspectives. A first-person perspective is provided which creates the impression that you are peering out over the steering wheel through the windshield. Other third-person perspectives are also provided where you are looking down at the race car from outside. The algorithms used to create these view points in a three-dimensional, 3-axis, format require an enormous amount of arithmetic horsepower.

The efficient arithmetic architecture, high level of integration and large on-chip memory of the SHARC processor has enabled SEGA to create an arcade game with superior performance and affordable price SEGA is continuously working to improve the performance of its system to make it the industry standard. ❖

Hypersignal Supports BittWare DSP Boards

Hyperception, Inc. announces support of Analog Devices' ADSP-21020 and ADSP-2106x (SHARC) EZ-LAB Development Boards and BittWare Research System's full line of PC-platform boards with real-time drivers and function blocks for its Hypersignal® for Windows Block Diagram visual DSP design product. This powerful combination of DSP hardware and visual programming software allows engineers to develop real-time signal processing applications for Analog Devices floating-point DSPs using a state-of-the-art graphical interface.

The release of the EZ-LAB board drivers from Hyperception provides a unique visual development environment that removes all of the inherent complexities of making efficient use of the hardware features of the DSP boards. Block Diagram's real-time support streamlines the algorithm design and test process by eliminating the code creation and debug steps. Users can get a prototype system up and running in a matter of minutes, and focus almost entirely on developing and fine-tuning their application-specific algorithms.

For a free evaluation package contact:

Hyperception, Inc.
9550 Skillman LB 125
Dallas, TX 75243
Tel: (214) 343-8525
Fax: (214) 343-2457
BBS: (214) 343-4108
Email: info@hyperception.com ❖

SHARCPAC Modules and Software from Alex Computer Systems

Typical SHARCPAC processing modules integrate one to eight SHARC processors with or without local SRAM or DRAM. Application specific SHARCPACs offer A/D conversion, digital I/O, and camera interfaces. Standard motherboards are supported including ISA, VME, and PCI.

By integrating products based upon the SHARCPAC specification, development can start immediately with minimum risk. Advanced high performance parallel DSP systems can be integrated simply and cost effectively with processing performance scaling through additional SHARCPAC modules.

Alex Computer Systems' SHARC portfolio provides DSP engineers with a comprehensive range of commercial off-the-shelf products. The portfolio is based upon various SHARCPAC modules and the APEX real-time software environment, the key to making effective use of the massive processing power of the SHARC. APEX, Alex's Advanced Parallel Executive simplifies application development while retaining highly efficient, scaleable, and portable code. APEX includes a real-time kernel for SHARC, a communications library, a task level debugger, optimized libraries, and a performance profiler. By providing high level features such as multi-tasking and network wide communication, the environment shields the user from the complex details of the hardware.

For more information contact:

Alex Computer Systems
204 Babcock Hall
118 Prospect Street
Ithaca, NY 14850
Tel: (607) 277-0678
Fax: (607) 277-0682
Email: sales@alexusa.com
Internet: www.alexusa.com ❖

SHARC Simulation Model from Synopsys Logic Modeling

The model of the ADSP-2106x SHARC allows designers to simulate and debug their DSP systems and ASICs prior to committing to board layout and silicon. Design problems can be uncovered early in the design cycle, reducing the number of board and ASIC turns. And because the model is fully functional, users can execute real code on the ADSP-2106x SHARC during simulation even before system prototypes are available.

The full-functional model of the SHARC uses Logic Modeling's patented hardware modeling technology, which is uniquely suited to developing extremely accurate models of the most complex devices. Because the hardware model uses the actual device in the simulation, it represents all device functions. The hardware model will allow users to see the ADSP-2106x processor behave in simulation exactly as it will in a finished system.

The model of the SHARC takes advantage of the features of the ModelSource 3400 hardware modeling system. The high-performance MS-3400 system provides direct support for the SHARC with the device's phase-locked loop enabled. This model is available for use with over 40 commercial simulators.

For more information, please contact your local Synopsys sales office:

Tel: 1-800-346-6335 or
1-503-690-6900,

Email: modelinfo@synopsys.com

Also the Logic Modeling Directory is available on the World Wide Web at http:www.synopsys.com, provides an interactive, searchable database of all the simulation models available from Logic Modeling, including its SmartModel, SourceModel, and hardware modeling technologies. ❖

Up To Date

EZ-KIT Lite Gets Even Better

THE EZ-KIT Lite development system for the ADSP-2100 Family has been upgraded. We've improved the download capability within the user interface to work more seamlessly with code generated by the C compiler. We've also added demonstration software for G.728 speech compression as well as a number of other new demonstration programs. This update also includes the latest version of Analog Devices' Assembly Development Tools (Assembler, Linker, Simulator, and PROM Splitter).

The software upgrade is available in the EZ-KIT Lite section of Analog Devices' Bulletin Board System (BBS) (617-461-4258), ftp site (ftp.analog.com), and World Wide Web site (http://www.analog.com). ❖

The EZ-KIT Lite Board features the ADSP-2181 DSP which is code compatible with the ADSP-2100 Family.

ADA Compiler

Tartan's Ada compiler for the ADSP-21060/62 SHARC DSP will be demonstrated in the Analog Devices booth at GOMAC (Government Microelectronics Applications Conference). This conference takes place from March 18–21,1996 at the Hyatt Regency, Orlando, Florida. ❖

G.723.1 on the ADSP-2171 and ADSP-2181

DSP Group, Inc. (Santa Clara, CA), co-author of the ITU-T G.723.1 specification of the speech codec for Low Bit Rate Video Telephony, announces the implementation of this G.723.1 TrueSpeech 6.3/5.3 kbps speech compression algorithm on the Analog Devices' ADSP-2171 and ADSP-2181 DSPs. Source and object code will be available in March 1996. The G.723.1 algorithm is being ported by AshCan Engineering (San Jose, CA).

This algorithm is optimized for ADSP-2171/ADSP-2181 and runs full-duplex using 20 MIPS of the available 33 MIPS. It is fully compliant with the ITU-T G.723.1 specification.

DSP Group currently has available a proprietary TrueSpeech 8.5, an 8.5 kbps speeech compression algorithm. TrueSpeech 8.5 is built into Microsoft Windows® '95 and runs on all Analog Devices' ADSP-2100 Family DSPs using only 10 MIPS.

For more information on G.723.1 (TrueSpeech 6.3/5.3) and /or TrueSpeech 8.5 on the ADSP-2171/ ADSP-2181, please contact:

Sandra Huang, Ph.D.
DSP Group, Inc.
3120 Scott Blvd.
Santa Clara, CA 95054
Tel: (408) 986-4432
Email: shuang@dspg.com ❖

ICSPAT '96 Call for Papers

A call for papers is being conducted for ICSPAT '96 (the 7th International Conference on Signal Processing Applications & Technology and the Data Acquisition Conference—both being held in conjunction with DSP World Expo and Data Acquisition Expo. These conferences will be held in Boston, Massachusetts, October 7–10, 1996 at the World Trade Center.

For ICSPAT '96, please mail, fax, or email 400-word abstracts with suggested application area by April 22, 1996 to:

Megan Forrester
Miller Freeman, Inc.
600 Harrison Street
San Francisco, CA 94107
Tel: (415) 356-3391
Fax: (415) 905-2220
Email: dsp@mfi.com

For the Data Acquisition Conference, please mail, fax, or email 400-word abstracts with suggested application area by April 30, 1996 to:

Data Acquisition Conference Staff
Miller Freeman, Inc.
600 Harrison Street
San Francisco, CA 94107
Tel: (415) 356-3391
Fax: (415) 905-2220
Email: dsp@mfi.com

To be eligible for consideration, abstracts must include full name, address, telephone, fax, and email of each author. Email submissions should be in simple text format (not encoded). ❖

ADSP-2100/21000 Family Workshop Schedule

The ADSP-2100 Family and ADSP-21000 Family Workshops are a fast way to get started designing with Analog Devices' DSPs. The workshops last three days and include hands-on lab sessions where you'll work with the hardware and software development tools available to support your design effort.

To register, place an order with your local Analog Devices Sales Office or Distributor for the **System Development & Programming With The ADSP-2100 Family workshop** (part# ADDS–21XX–WKSHP) or **System Development & Programming With The ADSP-21000 Family workshop** (part# ADDS–210XX–WKSHP), and call the site contact listed in the schedule with your registration information. Class size is limited to facilitate maximum individual instruction and practical hands-on experience.

Upcoming workshops are scheduled for the following dates and locations.

- Atlanta, Georgia

  Apr. 10-12, 1996 (2100 Family)
  May 15-17, 1996 (21000 Family)

  To register, contact Gordon Cooper at (404) 263-3722

- Campbell, California

  Mar. 6-8, 1996 (21000 Family)
  May 15-17, 1996 (2100 Family)
  June 5-7, 1996 (21000 Family)

  To register, contact Colin Duggan at (408) 879-3037

- Norwood, Massachusetts

  Apr. 3-5, 1996 (2100 Family)
  Apr. 10-12, 1996 (21000 Family)
  June 12-14, 1996 (2100 Family)

  To register, contact Nelia Elias at (617) 461-3672 or (617) 461-3881

Current Software Releases

Product	Latest Release
ADSP-2100 Family Development Software	5.1
ADSP-21000 Family Development Software	3.2
ADSP-2101 EZ-LAB	EPROM 1.4
ADSP-2111 EZ-LAB	EPROM 1.4
ADSP-2171 EZ-LAB	1.0
ADSP-21020 EZ-LAB	2.0 (H/W)
ADSP-2106x EZ-LAB	3.09 (S/W); 1.0 (FW)
ADSP-2101 EZ-ICE	5.01
ADSP-2111 EZ-ICE	5.01
ADSP-2171 EZ-ICE	5.01
ADSP-2181 EZ-ICE	5.11
ADSP-21020 EZ-ICE	3.2
ADSP-2106x EZ-ICE	3.2

Each release of the software is shipped with a Release Note. This note describes the current version and provides information on any upgrades to the software. Please be sure to return the registration card enclosed with your shipment! This allows us to keep you informed of subsequent releases.

Circle corresponding numbers on the enclosed reply card to request items of interest. The numbers appearing in parenthesis correspond to our Faxcode numbers.

1 ADSP-2100 Family DSP Microcomputers† (1579)
2 ADSP-2171/72/73 DSP Microcomputer (1869)
3 ADSP-2181 DSP Microcomputer (1927)
4 ADSP-21msp58/59 DSP Microcomputer (1901)
5 ADSP-21020 Floating-Point DSP Microprocessor (1578)
6 ADSP-21060/62 SHARC (1870)
7 ADSP-21csp01 Concurrent Signal Processor (Preliminary) (1975)
8 ADSP-2100 Family Development Tools (1919)
9 ADSP-21000 Family Development Tools (1489)

10 Selecting a DSP Processor (ADSP-2115 vs. TMS320C5x)
11 Selecting a DSP Processor (ADSP-2181 vs. TI & Motorola)
12 Selecting a DSP Processor (ADSP-21060 vs. TMS320C40)
13 Digital Signal Processing Brochure
14 System Development & Programming with ADSP-2100 & ADSP-21000 Families Workshop Brochure
15 EZ-KIT Lite Brochure
16 DSP/MSP Products Reference Manual

(Manuals must be ordered through your local sales office.)

- ADSP-2100 Family User's Manual (Order with part number ADSP-21XX-USERS-ML)
- ADSP-21020 User's Manual
- ADSP-2106x SHARC User's Manual
- ADSP-2100 Family EZ TOOLS Manual (Order with part number ADSP-21XX-EZ-MAN)
- ADSP-2100 Family Assembler Tools & Simulator Manual (Order with part number ADSP-21XX-DSW-ML)
- ADSP-2100 Family C Tools Manual (Order with part number ADSP-21XX-CTOOL-ML)
- ADSP-2100 C Runtime Library Reference (Order with part number ADSP-21XX-CRTL-MAN)
- ADSP-21000 Family Assembler Tools & Simulator Manual (Order with part number ADSP-210XX-DSW-MAN)
- ADSP-21000 Family C Tools Manual (Order with part number ADSP-210XX-CTOOLML)
- ADSP-21000 C Runtime Library Reference (Order with part number ADSP-210XX-CRTL-ML)
- Digital Signal Processing Applications Using the ADSP-2100 Family (Prentice Hall), Vol. 1 (Order with part number ADSP-21XX-APPS-BK1)
- Digital Signal Processing Applications Using the ADSP-2100 Family (Prentice Hall), Vol. 2 (Order with part number ADSP-21XX-APPS-BK2)
- ADSP-21000 Family Applications Handbook, Vol. 1

G2103-57-3/96

DSPatch® is published by the Computer Products Division of Analog Devices. It is created and edited by the Applications Group to aid those using or considering Analog Devices DSP components.

DSPatch welcomes comments, letters and articles on topics of interest to those designing digital signal processing and related high-performance systems. We will publish submitted articles that are of broad interest.

Additional copies are available from your local Analog Devices Sales Office. Address all correspondence to:

DSPatch
Analog Devices, Inc.
Computer Products Division
One Technology Way
P.O. Box 9106
Norwood, MA 02062-9106

You may also contact the Computer Products Division:

- *By contacting your local Analog Devices Sales Representative*
- *For Marketing information, call (617) 461-3881 in Norwood, Massachusetts, USA*
- *For Applications Engineering information, call (617) 461-3672 in Norwood, Massachusetts, USA*
- *The Norwood office Fax number is (617) 461-3010*
- *The Norwood office may also be reached by email at cpd_support@analog.com*
- *World Wide Web at HTTP://www.analog.com*

8 data bits, no parity, 1 stop bit, 300/1200/2400/9600/14400 baud

Type: ftp ftp.analog.com or ftp 137.71.23.21
Login as anonymous using your email address for your password

Call weekdays 8:00 a.m. to 5:00 p.m. Eastern Time
Applications Assistance is also available in:
Atlanta, Georgia at (404) 263-3722;
Campbell, California at (408) 879-3037; &
St. Louis, MO at (314) 921-4429.
Technical Assistance can also be obtained from Analog Devices Authorized Distributors